Atty Docket No. 74451.P115 *Patent*

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In re Application of:	
Jonathan J. Hull, et al.	Examiner: Ke, Peng
Application No.: 09/532,412	Art Unit: 2174
Filed: March 22, 2000	Confirmation No.: 8317
For: MELDED USER INTERFACES	

Mail Stop Appeal Brief- Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

REPLY BRIEF UNDER 37 C.F.R. § 41.41

Pursuant to 37 C.F.R. § 41.41, Appellant is filing this reply brief which addresses certain of the Examiner's points of arguments which were raised in the Supplemental Examiner's Answer dated October 28, 2008.

REMARKS

1. Stucka fails to teach extracting data of an application from a display buffer, particularly, without cooperation of the application

In the Examiner's Answer, the Examiner contends that Stucka's UIS (user interface server) loaded in a RAM (random access memory) "constantly extracts and stores interface elements that are modified by the user from the RAM, when it operates as a working memory area and as a display buffer" (10/28/2008 Examiner's Answer, page 11, emphasis added).

Appellant respectfully disagrees. It is respectfully submitted that the UIS loaded within the RAM of Stucka is not the same as a display buffer required by the present invention as claimed.

Rather, Stucka discloses a user interface server (UIS) that provides user interface services to multiple applications in a form of a library via a specific API (application programming interface), similar to a software development kit (SDK), which is used by a developer of the respective application. Specifically, Stucka states:

> "A user interface server coupled to applications, a display object store, and a window management system. The user interface server allows an application developer to provide an application with a user interface that is independent of any particular window management system."

(Abstract of Stucka, emphasis added).

Thus, Stucka discloses how an application communicates with the UIS via a set of APIs in order to use common functions provided by the UIS, which must be compiled and linked (via a compiler and linker of a software development tool, such as a C/C++ compiler and linker) with the application when the application is developed by the developer, to access one or more functions provided by the UIS. See, for example, col. 20, line 33 to col. 21, line 40, and col. 22, line 58 to col. 23, line 55.

However, it is respectfully submitted that Stucka does not disclose the limitation of extracting data of an application <u>from a display buffer</u>, particularly, without cooperation of the application. In fact, there is no mention of extracting data from a display buffer in Stucka.

Even if, for the sake of argument, the access from an application to a UIS server may be considered “extracting data” from a first application (assuming the UIS server is considered as a first application), the access to the UIS cannot be performed via a display buffer. As shown in Figure 3 of Stucka, the UIS is a server that is a dedicated server providing specific services over a network connection (e.g., connection 56 of Figure 3).

UIS 48 of Stucka is not stored in <u>a display buffer 72</u>. Rather, UIS 48, as well as other components, such as working memory area 78, display object store 46, and operating system 74, etc., is stored in a general purpose RAM 38. If Stucka intended to have UIS 48 operating within a display buffer, UIS 48 would have been shown within display buffer 72. There is no suggestion of storing UIS 48 in display buffer 72 throughout Stucka.

One with ordinary skill in the art would not attempt to store, even if it were possible, the UIS (e.g., a server), specifically the library, such as display object store 46, in a display buffer accessed by another application, such as application 50. It is respectfully submitted that one with ordinary skill in the art would not believe a server or a library, such as display object store 46 of Stucka, would reasonably be stored in a display buffer.

It is well known in the art that a display buffer is used to store graphics data (e.g., raster pixel data) representing a display output of an application prior to be or currently being displayed in a display device, which is completely different from data as a result of functional calls between an application and a server/library.

Even if the communications between applications (50, 53, and 54) and UIS 48 could be considered as extracting data from another application. Such an extraction operation is not performed from a display buffer, particularly display buffer 72. At most, it can only be considered an extraction from RAM 38. See, for example, Fig. 2, col. 7, line 47 to col. 8, line 25 of Stucka. Given the fact that Stucka fails to disclose or suggest that such operations be performed within display buffer 72, it is respectfully submitted that those with ordinary skill in the art would not consider, based on the teachings of Stucka, that the above operations can be performed within a display buffer.

In addition, the access to the UIS cannot be performed without cooperation of the UIS. As described above, in order to access UIS, an application has to be compiled and linked with the APIs of the UIS. Without using the APIs of the UIS, an application cannot access the UIS.

2. <u>Stucka fails to teach recognizing a layout from the first application</u>

The Examiner further contends that Stucka recognizes, stores, and applies every modification made by a user in working memory area, where the modifications include general layout component, such as background color, foreground color, text, font, etc. (10/28/2008 Examiner's Answer, page 11).

Appellant respectfully disagrees. Stucka does not disclose or suggest recognizing a layout from the data extracted <u>from a display buffer</u>, which is generated by another application. In fact, there is no mention of recognizing a layout in a display buffer in Stucka.

As discussed above, Stucka relies on a set of APIs used by an application to access a server to retrieve user interface data. There is no need to recognize a layout from the data provided by the UIS, particularly using a pattern recognition operation. The respective application has to be compiled and linked with the corresponding API, such as the header files and libraries of the UIS, during the development of the application. At run time, when the application calls a specific function (e.g., an API function) to access the server, the server provides services to the application. It is irrelevant whether the application would recognize a layout of the data returned from the server because the application has to call the corresponding function exported and published by the server in a format required by the API, regardless of whether the application recognizes the layout.

In contrast, the present invention as claimed recognizes a layout of a user interface generated by another application in a display buffer (e.g., raster pixel data that is about to be displayed or currently being displayed) and overlays its graphics data with the recognized interface, without using cooperation of the application generating the user interface. Here, the present invention as claimed is related to overlaying a display output of one application with the display output of another application. There is no disclosure or suggestion within Stucka that UIS overlays display outputs generated from multiple applications.

3. Stucka fails to teach creating an overlay for an application interface based on a recognized layout

The Examiner also contends that "Stucka creates an overlay for an application interface, because she teaches loading recognized interface components to an application interface" (10/28/2008 Examiner's Answer, page 12).

Appellant respectfully disagrees. As discussed above, an application of Stucka has to rely on a set of APIs (e.g., cooperation) to access the UIS in order to create a user interface. In fact, in order to access the UIS, an application of Stucka must be compiled and linked with libraries associated with the UIS at a development time. Nowhere in Stucka disclose or suggest an operation of creating an overlay of a recognized layout when an application accesses the UIS via a set of APIs. There is no need or motive to create an overlay from the display output of one application to display the display output generated from another application as discussed above, particularly, within a display buffer.

In contrast, the present invention as claimed includes using the recognized layout generated from a first application to create an overlay to display a second data generated by a second application, wherein there is no direct link between the first and second applications (e.g., without requiring communications between the first and second applications via APIs).

4. Stucka fails to teach first data extracted from the display buffer without cooperation of the first application at runtime

The Examiner further contends that "Stucka teaches extracting the display data without the cooperation of the application at runtime, because Stucka extracts the display without application volunteering the data to the working memory area for the purpose of recognizing the display layout" (10/28/2008 Examiner's Answer, page 12).

Appellant respectfully disagrees. As discussed above, an application of Stucka has to rely on a set of APIs (e.g., cooperation) to access the UIS in order to create a user interface. Nowhere in Stucka disclose or suggest an operation of creating an overlay of a recognized layout when an application accesses the UIS via a set of APIs. There is no need or motive to create an overlay to display the data from another application, as discussed above.

Further, contrary to the present invention as claimed, it is respectfully submitted that there is a direct link between the application and the UIS in Stucka. The source code of an application in Stucka <u>must be compiled and linked</u> with the exported or published APIs from the UIS. Any changes in the APIs would cause the communications between the application and the UIS to fail, which requires either the application or the UIS, or the both to be recompiled and linked. The applications and the UIS of Stucka depend on a set of APIs mutually agreed upon, which teaches away from the present invention as claimed. See, for example, col. 23, lines 38 to 54. Therefore, there is a direct link between an application and the UIS in Stucka. In contrast, there is no direct link between the first and second applications in the present invention as claimed, where the data is extracted without using an API of the first application.

With respect to Examiner's other points of arguments, Appellant respectfully maintains the arguments set forth in the appeal brief previously submitted.

For the reasons stated above, claims 1-40 are not anticipated under 35 U.S.C. §102(b) by Stucka and are patentable under 35 U.S.C. § 103(a) over Stucka in view of Kahl. Appellant respectfully prays for reversal of the Examiner's rejections.

Please charge Deposit Account No. 02-2666 for any shortage of fees in connection with this response.

Respectfully submitted,

BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP

Dated: December 29, 2008 /Kevin G. Shao/

Kevin G. Shao

kevin_shao@bstz.com
Attorney for Appellant
Registration No. 45,095

Customer No. 008791
1279 Oakmead Parkway
Sunnyvale, California 94085-4040
(408) 720-8300